Claims

What is claimed is:

[c1] A distributed system having a client and a server, comprising:

a state manager interposed between the client and the server, the state manager having a capability to generate a list of data attributes required to represent a state of the distributed system and a capability to cache data attributes so as to be locally accessible by the client; and

a service component interposed between the state manager and the server, the service component having a capability to fetch data from the server based on the list of data attributes.

[c2] The distributed system of claim **1**, further comprising a transport mechanism interposed between the state manager and the service component, the transport mechanism having a capability to package data for transport between the state manager and the service component.

[c3] The distributed system of claim **1**, wherein the state manager comprises means for learning data attributes required to represent a state of the distributed system.

[c4] The distributed system of claim **1**, wherein the state manager comprises means for creating a proxy for data in the server, the proxy having a capability to cache attributes of data.

[c5] The distributed system of claim **4**, wherein the state manager further comprises means for tracking changes made to attributes cached in the proxy.

[c6] The distributed system of claim **5**, wherein the state manager further comprises means for generating a list of attributes changed in the proxy.

[c7] The distributed system of claim **6**, wherein the service component comprises means for updating data using the list of attributes changed in the proxy.

[c8] The distributed system of claim **1**, wherein the state manager further comprises means for generating an executable instruction comprising a set of method calls to be executed on the server.

[c9] The distributed system of claim **8**, wherein the service component comprises means for interpreting the executable instruction.

[c10] The distributed system of claim **1**, wherein data comprises an object.

[c11] A distributed performance optimizer for a distributed application, comprising:

a client portion that generates a list of attributes of remote data required to represent a state of the application and that has a capability to cache attributes from the remote data; and

a server portion that fetches attributes from the remote data.

[c12] The distributed performance optimizer of claim **11**, further comprising a transport means that packages the attributes for transport between the server portion and the client portion.

[c13] The distributed performance optimizer of claim **12**, wherein the client portion comprises a plurality of proxies for the remote data in the distributed application, the proxies having a capability to cache attributes from the remote data.

[c14] The distributed performance optimizer of claim **13**, wherein the client portion further comprises means for tracking changes made to attributes cached in the proxies.

[c15] The distributed performance optimizer of claim **14**, wherein the service component comprises means for synchronizing the remote data with the proxies.

[c16] The distributed performance optimizer of claim **12**, wherein the client portion further comprises means for collecting information about attributes accessed in the proxies.

[c17] The distributed performance optimizer of claim **12**, wherein the client portion further comprises means for generating an executable instruction comprising a set of method calls to be executed on the server.

[c18] The distributed performance optimizer of claim **17**, wherein the service portion comprises means for invoking the executable instruction.

[c19] The distributed performance optimizer of claim **11**, wherein data comprises an object.

[c20] A method for optimizing a distributed application having a client and a server, comprising:

for each state of the application, predicting a set of data in the server and a set of corresponding data attributes required to represent the state of the application;

creating a proxy for each data in the set of data;

prefetching data from the set of data based on the set of corresponding data attributes; and

caching data in the proxy.

[c21] The method of claim **20**, wherein predicting the set of data comprises collecting information about data attributes accessed by the client for each state of the application and using the collected information to predict the set of data.

[c22] The method of claim **21**, wherein collection information comprises intercepting correspondence between the client and the proxies in order to determine data attributes accessed by the client.

[c23] The method of claim **21**, wherein collecting information about data attributes comprises determining a transition between a current state of the application and the next state of the application.

[c24] The method of claim **23**, wherein the transition comprises at least one method call from the client to be processed in the server.

[c25] The method of claim **24**, further comprising generating an executable instruction invoking the method call.

[c26] The method of claim **25**, further comprising executing the executable instruction on the server.

[c27] The method of claim **20**, wherein data comprises an object.

[c28] A method for optimizing an existing distributed application having a client and a server, comprising:

interposing a distributed performance optimizer between the client and the server so that correspondence between the client and the server is routed through the distributed performance optimizer;

creating a proxy for data in the server and making the proxy locally accessible to the client;

predicting a set of data attributes to fetch into the proxy for a current state of the application;

fetching the predicted set of data attributes from the server and storing data attributes fetched from the server in the proxy; and

synchronizing data attributes stored in the proxy with data attributes in the server.

[c29] The method of claim **28**, wherein predicting a set of data attributes to fetch comprises learning how the client interacts with the proxy.

[c30] The method of claim **28**, further comprising determining a set of method calls required to move from the current state of the application to a next state and generating an executable instruction invoking the set of method calls.

[c31] The method of claim **30**, further comprising executing the executable instructions on the server.

[c32] The distributed system of claim **28**, wherein data comprises an object.

[c33] A method for optimizing an existing distributed application having a client and a server, comprising:

generating a local representation of the server that is accessible to the client;

prefetching data from the server into the local representation to represent a state of the application; and

tracking the changes made to data fetched into the local representation; and

synchronizing data in the server with data in the local representation.

[c34] The method of claim **33**, wherein prefetching data from the server comprises prefetching data in a roundtrip between the local representation and the server.

[c35] The method of claim **34**, wherein prefetching data from the server comprises learning how the client interacts with the local representation of the server.

[c36] The method of claim **33**, wherein the local representation comprises a proxy for data in the server.

[c37] The method of claim **33**, wherein synchronizing data in the server comprises sending the changes made to data in the local representation and updating the server with data in a roundtrip between the local representation and the server.